APELLIDOS:	***NOMBRE:***
TITULACIÓN: __GII __GII+ADE	***DNI:***
Duración: 1:45 h.	
Calificación:	

Se recuerda que, según la guía docente, el total de la nota llegada la convocatoria de Junio se corresponde con la presente prueba escrita. Contestar en la hoja de enunciados los 4 primeros ejercicios.

1) (Prueba escrita teoría, 0.5 puntos) Enumera y describa brevemente los motores más importantes en la evolución de los lenguajes de programación de alto nivel imperativos.

2) (Prueba escrita teoría, 1 punto) Analice y calcule el orden de complejidad del siguiente pseudocódigo como se ha realizado en clase. La operación OrdenarListaEstática utilizará el algoritmo de *mergesort* u *ordenación por mezcla* para el que se creará su ecuación de recurrencia y se resolverá por el método de expansión de recurrencias.

```
DESDE i=1 HASTA 10 HACER
  DESDE j=1 HASTA 20 HACER
    DESDE k=1 HASTA 30 HACER
       OrdenarListaEstatica(l);
```

3) (Prueba escrita teoría, 1 punto) Especificar algebraicamente, tal y como se ha visto en clase, la sintaxis y la semántica de las operaciones Inorden y Frontera de árboles binarios. Utilice las operaciones de la especificación de árboles binarios u otros TADs vistos en clase que necesite.

4) (Prueba escrita práctica, 3.5 puntos) Defina los tipos de datos para representar en una única unidad de PASCAL un grafo estático y en otra unidad un grafo dinámico. A continuación escriba el código necesario de la unidad de grafo dinámico para pasar un grafo de su representación estática a una dinámica. Para ello considere disponibles las operaciones primitivas del TAD Grafo que necesite de la unidad con la representación estática indicando la entrada y salida de cada uso.

5) (Prueba escrita práctica, 4 puntos, MINIMO 1 PUNTO) PACTOS ELECTORALES

Tras los comicios del país de Nunca Jamás, y con resultados de los candidatos muy cercanos unos de otros, los Nuncajamasienses decidieron que era hora de nombrar gobiernos en los que se intentara reflejar la diversidad del pueblo.

El partido ganador, aunque lo hubiera hecho por poco, decidiría un 50% de los representantes del pueblo, el segundo la mitad (25%), el tercero la mitad del segundo (12,5%) y así sucesivamente hasta llegar al último partido con representación (fuera el porcentaje que fuera). Para ello se llegó al acuerdo de que hubiera 1 representante por cada 100.000 ciudadanos. Los datos del censo de Nunca Jamás están contenidos en una estructura arborescente ordenada por DNI y que de un año a otro cambia, puesto que el número de ciudadanos lo hace en función de los nacidos y fallecidos (aunque son muchos más de 100.000 ciudadanos).

a) [1 punto] Definir los tipos de datos necesarios para realizar este apartado. Dado el censo se creará una "lista de partidos" dinámica capaz de guardar la información de todos los representantes en el gobierno que tendrá cada partido empezando por el partido ganador y terminando con el de menor representación (se puede redondear a la baja dicho número de representantes mediante la función *floor*). La información que se guardará en cada nodo será la correspondiente de cada representante, su nombre y su posición en la "lista de su partido" (será un valor consecutivo para la sublista del mismo partido que se resetea de un partido a otro). Como siempre, se valorará una adecuada subprogramación.

b) [1 punto] De la "lista de partidos" se desea poder elegir una "lista de partido" que contenga solamente a los representantes de ese partido a partir del puesto *p* en el que ha quedado dicho partido en las elecciones. A cada nodo de la sublista se le agregará el mismo valor *p*, que representa la posición de su partido en la "lista de partidos".

c) [1 punto] Se desea poder crear una estructura llamada en inglés *heap* o montículo a partir de la estructura de lista de un partido. Este montículo de mínimos es un árbol binario en el que la información de la raíz tiene un valor menor que sus hijos, que a su vez cumplen la misma condición, de manera que en las hojas estarán los valores más grandes. Además, estos árboles se van llenando de izquierda a derecha completando los niveles del mismo. En este caso, la raíz del montículo contendrá al representante más importante del partido (el que presenta la menor posición en la lista de ese partido, el cabeza de lista) y por debajo estarán los siguientes más importantes. Este árbol será completo de izquierda a derecha de manera que el primer elemento de la lista será la raíz, los 2 siguientes sus hijos, los 4 siguientes los nietos, de manera que para el candidato k, sus hijos estarán en las posiciones 2k+1 y 2k+2, respectivamente (por ejemplo, para la raíz (k=0), sus hijos están en las posiciones 1 y 2). (VER FIG. 1). Definir el tipo de dato del montículo ~~dinámico~~ ESTATICO suponiendo que un representante tiene acceso a sus hijos y a su padre, y hacer los subprogramas necesarios para la creación de un *heap* a partir de la posición de un partido político en la "lista de partidos" del apartado a).

d) [1 punto] Por último, en las reuniones entre partidos se colocan en una mesa con un tablero para cada partido formando una figura geométrica parecida a una estrella en la que todos los cabeza de lista de los partidos se sienten en una esquina y puedan verse la cara para tratar los problemas de sus ciudadanos. Se desea modelar esta configuración a través de un grafo de políticos tal y como se describe en la FIG. 2. Definir los tipos de datos de la mesa y hacer un procedimiento tal que dado un puntero que apunta a un representante cualquiera, pueda encaminar un mensaje secreto a otro representante cualquiera de su lista o de cualquier otra, dados 2 valores enteros, que representan la posición del partido político empezando por el mayoritario y el puesto del representante de destino en su partido, y que devuelva un puntero apuntando a la posición del destinatario.

FIG 1. ESTRUCTURA DE PARTIDO CON UN CABEZA DE LISTA

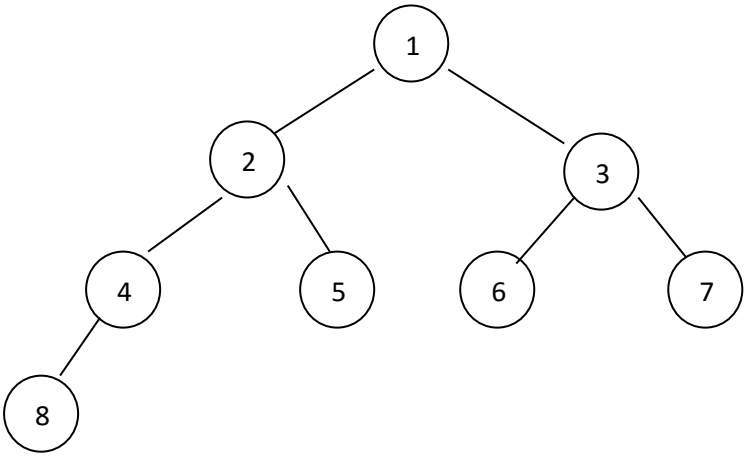

FIG 2. MESA REDONDA (POR CLARIDAD NO SE HAN REPRESENTADO TODOS LOS ARCOS)